

★ ColdBox Controller Common Methods

- **getColdboxOCM()**
Gets a reference to the object cache manager
- **getColdboxSettings()**
Gets the framework settings structure
- **getConfigSettings()**
Gets your application's settings structure
- **getSetting(name, [framework=false])**
Gets a setting key from either structure
- **setSetting(name, value)**
Sets a new setting in the application's structure
- **settingExists(name, [framework=false])**
Checks whether the named setting exists.
- **getSettingStructure([framework=false], [deepcopy=false])**
Gets the settings structure.
- **getPlugin(name , [customPlugin=false], [newInstance=false])**
Get's a core plugin or custom plugin
- **getMyPlugin(name)**
Get's a custom plugin
- **runEvent([event=DefaultEvent], [prepostExempt=false])**
Execute an event.
- **setNextEvent([event=DefaultEvent],[queryString=""])**
Relocate to another event.
- **throw(message, [detail=""],[type='framework'])**
Cfthrow façade

★ ColdBox Object Cache Manager Common Methods

- **Lookup(key)**
Looks up an object in the cache
- **Get(key)**
Gets an object from cache.
- **Set(key, object, timeout)**
Set an object in the cache. Timeout is in minutes. If 0, then it is permanently in the cache.
- **clearKey(key)**
Cleans an object from the cache
- **expireAll()**
Expires all objects
- **Reap()**
Runs the internal object cleaner

★ ColdBox Handlers/Plugins Common Methods

- **getController()**
Get the ColdBox controller reference
- **getDatasource(alias)**
Create and return a datasource bean object according to the alias sent in.
- **getMailSettings()**
Create and return a mail settings bean according to your config.xml.cfm
- **renderView(view)**
Façade to the renderer plugin.
- **getfwLocale()**
Get the default locale string if using i18n
- **getResource()**
Returns a resource from a resource bundle if using i18n
- **includeUDF()**
Injects a UDF template into the handler/plugin

★ Sessionstorage Plugin (Façade) Common Methods

- **getVar(name,[defaultValue=""])**
Get a variable from the session scope
- **setVar(name, value)**
Set a variable in the session scope
- **exists(name)**
Checks if a value exists in the session scope
- **deleteVar(name)**
Tries to delete a variable in the session scope.

★ Clientstorage Plugin (Façade) Common Methods

- **getVar(name,[defaultValue=""])**
Get a variable from the client scope
- **setVar(name, value)**
Set a variable in the client scope
- **exists(name)**
Checks if a value exists in the client scope
- **deleteVar(name)**
Tries to delete a variable in the client scope.

★ IoC Plugin Common Methods

- **getBean(name)**
Get a bean from the IoC framework, if IOCObjectCaching is used, then it will cache objects in the OCM.
- **getIoCFactory()**
Gets the factory object reference.
- **getIoCFramework()**
Gets the name of the framework being used.
- **Configure()**
Reload the configuration file and re-creates the factory.
- **reloadDefinitionFile()**
Reloads the IoC factory with the definition file.

★ Messagebox Plugin Common Methods

- **setMessage(type, message)**
Set a new message. Type = info,warning,error
- **getMessage()**
Returns a structure of the type and message
- **clearMessage()**
Clears the message
- **isEmpty()**
Checks if there is a message set
- **Renderit()**
Renders the messagebox.

★ Renderer Plugin Common Methods

- **renderView(view [default=""])**
Returns the rendered view if passed, else renders the view set in the request collection. The event argument is mandatory, it's the request collection
- **renderExternalView(view)**
Bring back the rendered external document.
- **renderLayout()**
Render the layout/view combo

★ Webservices Plugin Common Methods

- **getWS(name)**
Gets the WSDL Url according to name
- **getWSobj(name)**
Returns an instantiated web service object
- **refreshWS(name)**
Refresh the webservice's stubs

★ Logger Plugin Common Methods

- **logEntry(Severity,Message, [ExtraInfo=""])**
Log an entry into the log files.
- **logError(Message,Exception,[ExtraInfo=""])**
Explicitly log an error.
- **logErrorWithBean(ExceptionBean)**
Explicitly log an error with an ExceptionBean
- **tracer(Message,[ExtraInfo=""])**
Trace messages to the debugger Panel

★ BeanFactory Plugin Common Methods

- **populateBean(FormBean)**
If passing a base path, it will create and populate the bean with the contents of the request collection. If passing an instantiated bean (java/cfc) it will populate it with the request collection as signatures are matched.

★ queryHelper Plugin Common Methods

- **filterQuery(query, field, value, cfsqltype)**
Filter a query
- **sortQuery(query,sortBy, sortOrder)**
Sort a query.
- **getColumnArray(query,ColumnName)**
Returns an array of the values of the column passed in.
- **getCountDistinct(query,ColumnName)**
Returns total/count distinct values of the column.
- **getRowNumber(query,ColumnValue,ColumnName)**
Get's the row number of the first matched value.

★ Timer Plugin Common Methods

- **logTime(label, TickCount)**
Log a timer entry.
- **start(label)**
Start a labeled Timer.
- **stop(label)**
Stop the labeled Timer.

★ EVENT (RequestContext) Object

The event object is composed of a user's request data. It contains the FORM/URL and much more. Below are some common methods.

- **getCollection()**
Gets the reference to the internal collection. Useful for creating shortcuts in views/layouts
Example:

```
<cfset rc = event.getCollection()>
<!-- Insert new key-->
<cfset rc.NewKey = "My New Key">
```
- **collectionAppend(structure, [boolean overwrite=false])**
Append a structure to the collection with an overwrite flag.
- **getCurrentEvent()**
Gets the current incoming event.
- **overrideEvent(event)**
Override the current set event to execute with your override.
- **getCurrentLayout()**
Gets the current set layout
- **getCurrentView()**
Gets the current set view
- **getSize()**
The size of the collection (How many items)
- **getValue(name , [defaultValue])**
Get a value from the collection by name. You can also pass in a default value to return if the key is not found.
This can be a simple or complex variable. Or a multi structure name:
ie: myStruct.data.MyName
paramValue(name , value)
Similar to cfparam. It will param a value in the collection. If the key does not exist it will create it with the default value.
- **valueExists(name)**
Returns a boolean if the name exists in the collection.
- **removeValue(name)**
Remove a value from the collection
- **setValue(name, value)**
Set a value in the collection
- **setLayout(layout)**
Set/override a layout for the incoming request. (Do not append .cfm)
- **setView (name, [boolean nolayout=false])**
Set a view to render. The **nolayout** parameter is optional and it defaults to false. If set to true, view renders by itself, no layout.
- **showdebugPanel(boolean)**
Whether to turn on/off the debug panel for the incoming request, good for Ajax Requests.
- **getDebugPanelFlag()**
Gets the set debug panel flag.
- **getDefaultLayout()**
Get the default layout
- **getDefaultView()**
Get the default view.
- **getEventName()**
The name of the event variable. (event= or do=)
- **getViewLayouts()**
Get the view-layouts combo array

★ To re-init the Framework use 'fwreinit=true' in the URL

To go into debug mode use:
Index.cfm?debugmode=true&debugpass={debug password}

To Render the cache debugging panel use (You must be in debug mode ONLY):
Index.cfm?debugpanel=cache

Override the config file location by setting the COLDBox_CONFIG_FILE = 'path to coldbox.xml' before including the framework.

★ Application Layout

```
|AppName
|--> config
|--> handlers
|--> layouts
|--> logs
|--> model
|--> views
|--> index.cfm
```